Vyhledávání hodnoty

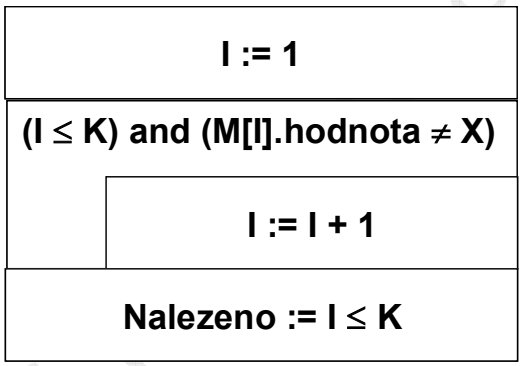
Sekvenční vyhledávání

Přímá metoda

- **Předpoklady:**
  - Vyhledávací prostor (datová struktura) **M** obsahuje **K** prvků **P**, jednotlivé prvky jsou přístupné indexem
  - Hledaná hodnota je **X**.

- **Popis metody:**
  - Postupně procházíme prvky s indexem **I = 1 .. K** a porovnáváme hodnotu prvku s hodnotou **X**
  - Hodnota **X** je nalezena v případě, že index **I ≤ K**

- **Implementace:**




```
begin
     I:=1;
     while (I<=K) and (M[I].hodnota<>X) do
          I:=I+1;
     Nalezeno:=I<=K;
end.
```

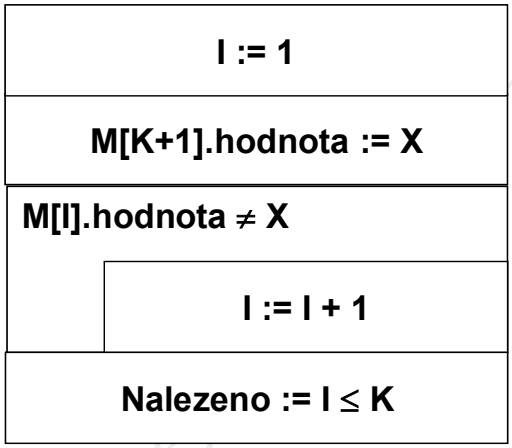
- **Operační složitost**
  - maximálně **K** opakování cyklu – **lineární složitost** (řádu *n*)

Se zarážkou

- **Předpoklady:**
  - Vyhledávací prostor (datová struktura) **M** obsahuje **K** prvků **P**, jednotlivé prvky jsou přístupné indexem
  - Hledaná hodnota je **X**.

- **Popis metody:**
  - Modifikace předchozí metody: **M[K+1] := X**
  - Postupně procházíme prvky s indexem **I = 1 .. K+1** a porovnáváme hodnotu prvku s hodnotou **X**
  - Hodnota **X** je nalezena v případě, že index **I ≤ K**

- **Implementace:**




```
begin
     I:=1;
     M[K+1].hodnota:=X;
     while M[I].hodnota<>X do I:=I+1;
     Nalezeno:=I<=K;
end.
```

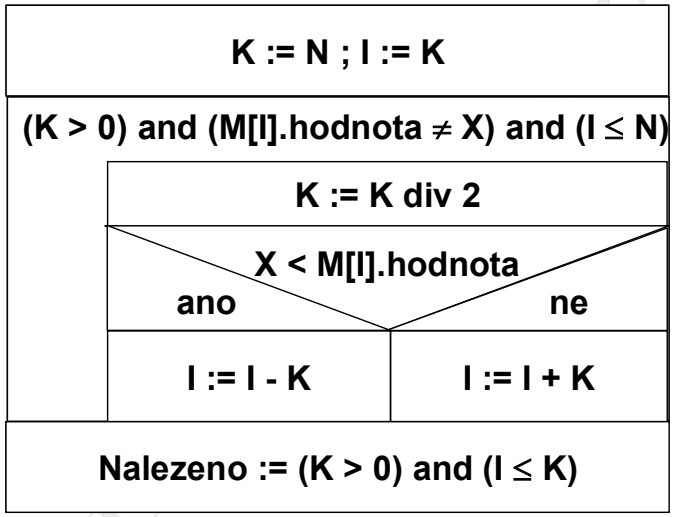
- **Operační složitost**
  - maximálně **K+1** opakování cyklu – **lineární složitost** (řádu *n*)
  - zjednodušení podmínky cyklu

Vyhledávání půlením intervalů

- **Předpoklady:**
  - Vyhledávací prostor (datová struktura) **M** obsahuje **N** prvků **P**, jednotlivé prvky jsou přístupné indexem
  - **Prvky jsou uspořádány vzestupně**
  - Celkový počet prvků je celistvou mocninou dvou.
  - Hledaná hodnota je **X**.

- **Popis metody:**
  - Nejprve zjistíme, zda není hledaným prvkem poslední
  - Porovnáme hledanou hodnotu **X** s prvkem uprostřed datové struktury
  - Je-li **X** menší než prostřední prvek, hledáme v polovině „menších“ prvků, jinak v polovině „větších“

- **Implementace:**




```
begin
     K:=N;I:=K;
     while (K>0) and (I<=N) and (M[I].hodnota<>X) do
     begin
          K:=K div 2;
          if X<M[I].hodnota then I:=I-K
          else I:=I+K;
     end;
     Nalezeno:=(K>0)and(I<=K);
end.
```

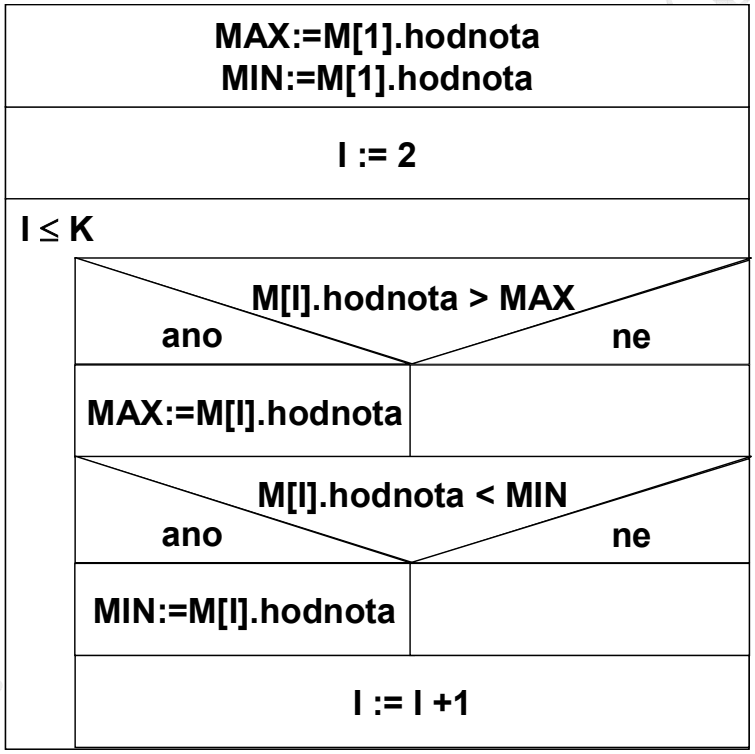
- **Operační složitost**
  - počet průchodů cyklem odpovídá funkci $\log_2 n$

Vyhledávání maxima / minima

- **Předpoklady:**
  - Vyhledávací prostor (datová struktura) **M** obsahuje **K** prvků **P**, jednotlivé prvky jsou přístupné indexem
  - Hledáme **maximální** a **minimální** hodnotu

- **Popis metody:**
  - Za maximum (**MAX**) i minimum (**MIN**) považujeme **první prvek**
  - Postupně procházíme prvky s indexem **I = 2 .. K** a porovnáváme hodnotu prvku s hodnotami **MAX** a **MIN**
  - Je-li hodnota prvku větší než maximum, stává se novým **maximem**
  - Je-li hodnota prvku menší než minimum, stává se novým **minimem**

- **Implementace:**




```
begin
      MAX:=M[1].hodnota;
      MIN:=M[1].hodnota;
      I:=2;
      while (I<=K) do begin
            if (M[I].hodnota>MAX) then MAX:=M[I].hodnota;
            if (M[I].hodnota<MIN) then MIN:=M[I].hodnota;
            I:=I+1;
      end;
end.
```

- **Operační složitost**
  - **K-1** opakování cyklu – **lineární složitost** (řádu *n*)